A Novel Approach To Integration By Parts Reduction

Robert M. Schabinger

based on arXiv:1406.4513 with Andreas von Manteuffel

The PRISMA Cluster of Excellence and
Mainz Institute of Theoretical Physics
Johannes Gutenberg Universität

Outline

Integration By Parts in d Dimensions

F. Tkachov, Phys. Lett. **B100**, 65, 1981; K. Chetyrkin and F. Tkachov, Nucl. Phys. **B192**, 159, 1981

$$\begin{aligned} 0 &= \int \frac{d^d\ell}{(2\pi)^d} \frac{\partial}{\partial \ell_\mu} \left(\frac{\ell_\mu}{(\ell^2 - m^2)^a} \right) \\ &= \int \frac{d^d\ell}{(2\pi)^d} \left(\frac{d}{(\ell^2 - m^2)^a} - \frac{2a\ell^2}{(\ell^2 - m^2)^{a+1}} \right) \\ &= (d - 2a) I(a) - 2am^2 I(a+1) \end{aligned}$$

$$\Longrightarrow I(a) = \frac{(-1)^a \Gamma(a - d/2)}{\Gamma(1 - d/2)\Gamma(a)\,(m^2)^{a-1}} I(1)$$

In general, one must consider all integration by parts relations generated by $\{\ell_1^\mu, \ldots, \ell_L^\mu\}$ AND $\{k_1^\mu, \ldots, k_N^\mu\}$ for each differentiation variable ℓ_j^μ and consider irreducible numerators.

Can We Solve These IBP Relations?

Suppose we want to solve the system of IBP recurrence relations to determine the master integrals for a given multi-loop topology:

- For most interesting examples a highly non-trivial system of recurrence relations results.
- Recurrence relations are typically hard to solve directly.
- A well-known algorithm due to Laporta (S. Laporta hep-ph/0102033) reduces the problem to a very large system of linear equations which can be solved using linear algebra.

Laporta's algorithm, while a major breakthrough, requires exorbitant computational resources for most interesting multi-loop topologies. What, if anything, can be done to improve the situation?

The Complexity of “Gaussian” Elimination

- It is widely believed that the computational complexity of Gaussian elimination is $\mathcal{O}\left(n^3\right)$ for $n \times n$ rational matrices.
- This is far too simplistic and is true only if each arithmetic operation takes essentially the same amount of time.
- In a finite field, the “grade school” algorithm does have $\mathcal{O}\left(n^3\right)$ complexity but, even over the rational numbers, the situation is much worse because the numerators and denominators of the rational numbers typically increase in size after every operation.
- Intermediate expression swell is a severe problem for the grade school Gaussian elimination algorithm and can lead to run-times and run-time storage requirements which are exponential in n!

X. G. Fang and G. Havas, ISSAC ’97, 28, (1997)

And It Gets Worse...

- The linear systems that one obtains from Laporta's algorithm will always have *polynomial* entries at the outset and this introduces additional complications.
- Avoiding unrecognized zeros during the course of the elimination procedure requires a very large number of polynomial greatest common divisor (GCD) computations.
- These operations actually account for a substantial fraction of the total run-time of most currently available integration by parts reduction codes.
- Intermediate expression swell manifests itself in the degrees of the polynomial numerators and denominators of the rational functions that appear at intermediate stages of the reduction.

And It Gets Worse...

- The linear systems that one obtains from Laporta's algorithm will always have *polynomial* entries at the outset and this introduces additional complications.
- Avoiding unrecognized zeros during the course of the elimination procedure requires a very large number of polynomial greatest common divisor (GCD) computations.
- These operations actually account for a substantial fraction of the total run-time of most currently available integration by parts reduction codes.
- Intermediate expression swell manifests itself in the degrees of the polynomial numerators and denominators of the rational functions that appear at intermediate stages of the reduction.

Luckily for us, an enormous amount of mathematical research has been devoted to ameliorating these problems!

The General Idea

- To save time, we restrict ourselves in this talk to linear systems with coefficients in $\mathbb{Q}[d]$. Let us stress, however, that a multivariate generalization has been worked out as well.
- Actually, it turns out that there is a very close connection between the problem of linear system solving over $\mathbb{Q}[d]$ and the seemingly simpler problem of linear system solving over $\mathbb{Q}$ since both $\mathbb{Q}[d]$ and $\mathbb{Q}$ are Euclidean domains.
- Map to "small" prime fields and then work over them for most of the calculation to avoid intermediate expression swell.
- Sew the solutions together by "Chinese remaindering."
- Reconstruct the true, rational coefficients of all polynomials which appear in the entries of the null space vectors.

The Extended Euclidean Algorithm

Begin with $(g_0, s_0, t_0) = (a, 1, 0)$ and $(g_1, s_1, t_1) = (b, 0, 1)$

$$\begin{aligned} q_i &= g_{i-1} \text{ quotient } g_i \\ g_{i+1} &= g_{i-1} - q_i g_i \\ s_{i+1} &= s_{i-1} - q_i s_i \\ t_{i+1} &= t_{i-1} - q_i t_i \end{aligned}$$

The algorithm terminates when $g_{k+1} = 0$ for some k. At that point

$$s_k\, a + t_k\, b = g_k = \mathrm{GCD}(a, b)$$

The Image Of A Rational Number Under The Canonical Homomorphism of $\mathbb{Z}$ onto $\mathbb{Z}_m$

Among other things, the extended Euclidean algorithm allows one to define multiplicative inverses in prime fields.

Given m and b such that $\mathrm{GCD}(m, b) = 1$, we see that

$$\begin{aligned} 1 &= s\, m + t\, b \\ \Rightarrow \quad 1/b &\equiv t \bmod m \end{aligned}$$

Denoting the canonical homomorphism of $\mathbb{Z}$ onto $\mathbb{Z}_m$ by $\phi_m(z)$,

$$\phi_m(a/b) = \phi_m(a)\phi_m(1/b)$$

What About Going The Other Way?

Mapping the coefficients of our polynomials to prime fields is not going to help unless we have some way to invert the map $\phi_m(z)$.

$$g_i = s_i m + t_i u \quad \text{for all } i$$

when one applies the extended Euclidean algorithm to u and $m > 0$.

If, as will usually be the case, $(m, t_i) = 1$ for all i,

$$g_i / t_i \equiv u \bmod m \quad \text{for all } i$$

which implies that, typically, $\phi_m^{-1}(z)$ cannot be defined.

Rational Reconstruction

P. S. Wang, SYMSAC '81, ACM Press, 212 (1981);

P. S. Wang *et. al.* SIGSAM Bulletin **16**, No. 2, 2 (1982)

- Remarkably, under appropriate conditions, the map $\phi_m(z)$ *does* have an inverse.
- For a given rational number, a/b, one can invert $\phi_m(z)$ if $m > 2\max\{a^2, b^2\}$.
- In this situation, the unique solution to the rational reconstruction problem is given by:

$$\frac{a}{b} = \frac{g_j}{t_j}$$

where g_j is the first g_i in the extended Euclidean algorithm to violate

$$|g_i| > \lfloor \sqrt{m/2} \rfloor$$

Linear System Solving Over $\mathbb{Q}$

We now have all the ingredients we need to describe a fast and memory-efficient algorithm for the solution of linear systems over $\mathbb{Q}$.

- On a computer cluster, select some number of cores, n_c, likely to be larger than the length of the nastiest integer expected in the result (measured in machine words).
- On each core, choose a largish machine-sized prime (*e.g.* $2^{64} - 59$), p_i, take the image of the linear system modulo p_i, and then solve the system n_c times in parallel.
- In this fashion, a solution, $\mathbf{k}_i$, is generated on each core modulo the corresponding p_i and these solutions can be sewn together using the Chinese remainder algorithm to produce a lifted solution, $\mathbf{K}$, modulo $p_1 \cdots p_{n_c}$.
- Finally, we can attempt a rational reconstruction on the coefficients. If the procedure succeeds we are done. Otherwise, we have to compute additional samples and then try once again.

see *e.g.* M. Kauers, Nuclear Physics B (Proc. Suppl.), **183**, 245 (2008)

Linear System Solving Over $\mathbb{Q}[d]$

The system solving algorithm for $\mathbb{Q}[d]$ is virtually identical to the algorithm for $\mathbb{Q}$ because both $\mathbb{Q}[d]$ and $\mathbb{Q}$ are Euclidean domains.

- Linear systems over $\mathbb{Z}_{p_i}[d]$ can be solved by sampling some number of times likely to be large relative to the worst-case total degree of the rational functions expected to appear in the result.
- Polynomial interpolation provides an analog of Chinese remaindering: $p_1 \cdots p_{n_c} \to (d - p_1) \cdots (d - p_{n_c})$.
- The rational reconstruction algorithm is essentially the extended Euclidean algorithm with a modified termination criterion. Its structure is such that one can immediately write down an analogous algorithm for univariate rational *functions*.

see *e.g.* M. Kauers, Nuclear Physics B (Proc. Suppl.), **183**, 245 (2008);

S. Khodadad and M. Monagan, ISSAC '06, 184, (2006)

Why You Should Take This Seriously

- Despite the maturity of the subject, it is surprisingly difficult to find a working public implementation. Manuel Kauers's `Mathematica` package `LinearSystemSolver.m` is the only example known to us but is rather impressive in its performance.
- For a relatively small system (387 equations in 92 unknowns) with coefficients in $\mathbb{Q}[d]$, we observe about a factor of 40 improvement over `Mathematica`'s `Solve` function *on a single core*.
- For the specific application of IBP reduction, substantial gains will still be possible with many scales because, in the modular approach, only the IBP relations relevant to the simplification of the physical amplitude need to be reconstructed. For most problems, often only a tiny fraction of the relations produced by a standard run of Laporta's algorithm are actually needed to reduce the physical amplitude down to master integrals.
- However, we also expect that some amount of intermediate expression swell will be unavoidable with many scales and that parallelization will become progressively more important.

Outlook

Overall, the modular paradigm has excellent prospects.
Just two drawbacks come to mind:

- Although we have in hand a feasible multivariate generalization of the method outlined above for $\mathbb{Q}[d]$, there is no guarantee that our approach to the reconstruction of multivariate rational functions is the best one. Other ideas exist and experimentation will likely be required to identify a nearly-optimal strategy.
- Implementing the ideas discussed in this talk will require a rather substantial rewrite of the `Reduze` code and care must be taken at every step with the implementation details so as not to spoil the performance of the algorithm.